Distance-based Kernels for Dynamical Movement Primitives

Diego ESCUDERO-RODRIGO [a,1], René ALQUEZAR [a],
[a] *Institut de Robotica i Informatica Industrial, CSIC-UPC, Spain*

Abstract. In the Anchoring Problem actions and objects must be anchored to symbols; and movement primitives as DMPs seems a good option to describe actions. In the bottom-up approach to anchoring, the recognition of an action is done applying learning techniques as clustering. Although most work done about movement recognition with DMPs is focus on weights, we propose to use the shape-attractor function as feature vector. As several DMPs formulations exist, we have analyzed the two most known to check if using the shape-attractor instead of weights is feasible for both formulations. In addition, we propose to use distance-based kernels, as RBF and TrE, to classify DMPs in some predefined actions. Our experiments based on an existing dataset and using 1-NN and SVM techniques confirm that shape-attractor function is a better choice for movement recognition with DMPs.

Keywords. trajectories, DMP, learning, kernel, classify, 1-NN, SVM, actions

1. Introduction

The control of robotic systems has reached a high level of precision and accuracy. However, high complexity and task specificity are limiting factors to robots that interact with humans or work in real life environments.

In these situations *generalist robots* that adapt to a novel situation are needed. These robots should act goal-oriented by making plans, actively explore their environment, and identify opportunities for actions. And they should improve their knowledge interacting with humans and through learning from experience. Generalist robots are composed by perception/actuation and knowledge representation systems. Therefore, they need that symbols in the high-level system are coupled to objects and actions of the low-level system. This sensor(actuator)-to-symbol problem is named as the *Anchoring Problem* [1]; and the way to create anchors depends on what we are trying to anchor: objects or actions, and how we are doing it: top-down or bottom-up.

Anchoring actions from movement primitives (a.k.a. units of actions, basis behaviors, etc.) seems a good choice, because they are compact parameterizations of the robot control policy and a well-established approach for representing modular and re-usable robot movement generators [2][3]. There are different types of movement primitives but *Dynamical Movement Primitive (DMP)* is the most suitable one for us to represent actions. That is due to its properties that will be explained in the next section. In the bottom-

[1] e-mail: descudero@iri.upc.edu

up approach to anchoring, new symbols may be associated with clusters in conceptual spaces [4], and to obtain such clusters, some distance or similarity measure between the concept representations has to be defined.

The paper starts describing the DMP framework and two different formulations. Afterwards, we focus on how to use them for movement recognition; analyzing the properties of the shape-attractor and the weights, and explaining the two distance-based kernels that we propose to use. Finally, we apply our classification approach to a trajectory dataset to test its reliability.

2. Understanding DMPs

A DMP is a model-free approach that encodes a one-dimensional trajectory as a set of differential equations, and these equations can compactly represent control policies [5]. The basic idea is that a dynamical system with well specified stable behavior is taken, and another term (the shape-attractor) is added that makes it follow some interesting trajectory. Two behavioral patterns, rhythmic or discrete, can be used with DMPs. In this paper we only deal with discrete ones, but our proposal could be easily applied to rhythmic ones as well.

A useful characteristic of the DMPs is that they could be learned from only one demonstration, making them a useful tool in *Learning by Demonstration* with robots. However, some researchers have extended it to be able to learn a DMP from multiple demonstrations. For example, in [6] an extension named as Stylistic Dynamic Movement Primitives (SDMPs) was defined.

Although the DMP framework was original defined for one-dimensional trajectories, it can be used with multi-dimensional ones, too. And as the framework is the same in both cases, we decided to name a DMP as single DMP (s-DMP) when we are using it with a one dimensional trajectory, and multi DMP (m-DMP) in the other case.

2.1. Original formulation

The original formulation [5] of a DMP is

$$\begin{aligned} \tau \dot{v} &= K(g - x) + (g - x_0)f(s) - Dv \\ \tau \dot{x} &= v \end{aligned} \tag{1}$$

The above dynamical system, also known as transformation, is composed of two main components [7] without considering the global damping term $-Dv$:

- The goal-attractor: $K(g - x)$, this term attracts x towards the goal position g.
- The shape-attractor: $(g - x_0)f(s)$, this term represents the contribution of the non-linear forcing term scaled by the $(g - x_0)$ factor.

The non-linear forcing term, f, is defined as

$$f(s) = \frac{\sum_{i=1}^{N} \psi_i(s) w_i}{\sum_{i=1}^{N} \psi_i(s)} s \tag{2}$$

and

$$\psi_i(s) = \exp(-h_i(s - c_i)^2) \tag{3}$$

where s exponentially decays from 1 to 0 with time ($\dot{s} = -\alpha s$), and w_i named as weights are free parameters to learn. Due to this formulation a DMP has the following properties:

- Any smooth movement can be generated with a DMP.
- The differential equations converge to the goal point g and automatically adapt to perturbations of the state x.
- The generated movement adapts online to a change of the goal g.
- The differential equations are translation invariant.
- The duration of a movement could be changed using τ without changing the movement trajectory.

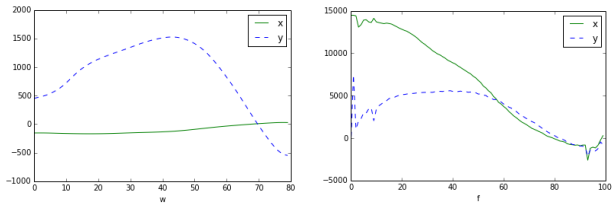

Figure 1. Example of learned w and f from a *C* letter trajectory using original formulation

On the other hand, this formalism has also some issues:

- If start and end-point of a movement are the same, $x_0 = g$, no movement will be generated, $v(t) = 0$.
- If g is close to x_0, a small change in g may produce huge accelerations that break the limits of the robot.
- If changing g across the zero point, the whole movement inverts.

2.2. Bio-inspired formulation

As some of the issues of the original formulation are problematic, a new bio-inspired formulation was introduced in [8].

$$\begin{aligned} \tau\dot{v} &= sK(\frac{f(s)}{s} + x_0 - x) + (1 - s)K(g - x) - Dv \\ \tau\dot{x} &= v \end{aligned} \tag{4}$$

In this formulation the two attractor terms are defined as:

- The goal-attractor: $K(g - x)$.
- The shape-attractor: $K(\frac{f(s)}{s} + x_0 - x)$, this term represents an attractor towards the moving point $\frac{f(s)}{s} + x_0$.

Main differences between the bio-inspired formulation and the original one are:

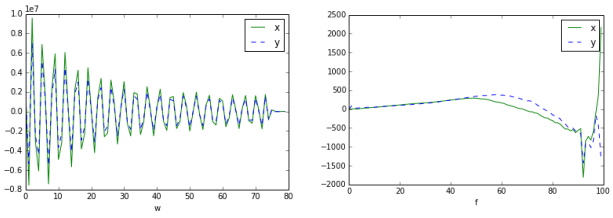

Figure 2. Example of learned w and f from a *C* letter trajectory using bio-inspired formulation

- It bypasses the issues arising when the goal is close to the origin of the trajectory.
- It improves the adaptation to new goals since the shape-attractor does not scale anymore with $(g - x_0)$.
- The addition of the x_0 component on the shape-attractor enables the system to behave properly when the initial starting point is changed.
- And it is invariant under rotations of the coordinate system.

Other modifications of the formulation, as weight decoupling [9], already exist but they are not interesting for our discussion.

3. Movement Recognition with m-DMPs

As previously said, we could use an m-DMP to characterize a multi-dimensional trajectory. In multi-dimensional trajectories an s-DMP is used for each trajectory component (or DOF) but there is only one shared canonical system that synchronize them.

The DMP is a framework that provides great learning and motion reproduction capabilities. And the invariance properties, that were explained in previous section, make it a good option to movement recognition. For example, these invariance properties render the parameters w insensitive to movement translation, rotation, and spatio-temporal scaling. In addition, other useful properties for movement recognition are: compactness, robustness and comparison efficiency. As their internal representation of a movement is compact, it consumes small memory and comparison between two s-DMPs can be achieved without requiring much new effort.

Ijspeert *et al.* [2] have shown that correlations between weight parameters w of the DMPs reflect similarities between motions; trajectories that are topologically similar tend to be fit by similar parameters w. However, this property does not hold for all formulations of the DMPs, since different equations have been proposed through time by several authors and some of the approaches seem not suitable for recognition.

In this section we analyze the properties of shape-attractor function f and weights w in the two DMP formulations recalled in previous section. This is done from movement recognition point of view and considering m-DMPs. Afterwards, two distance-based kernels which could be applied to the m-DMPs are reviewed.

3.1. Shape-attractor function vs. weights

To compare m-DMPs, either the weights or the shape-attractor function of all s-DMPs that compose these m-DMPs must be used.

The shape-attractor term encapsulates the learned trajectory and stimulates the system to mimic that trajectory. According to Eqs. (1) and (4) the shape-attractor is predominant at the beginning of the movement, when s is closed to 1; while its influence is small at the end of the movement, when s is closed to 0. The $f(s)$ function characterizes the spatio-temporal path of a DMP, and it is a normalized basis function representation with linear parameterization. And the weights w_i that determine the particular shape of the trajectories are included in $f(s)$, see Eq. (2).

As the weights w are obtained solving Eq. (1) or Eq. (4), their value will be different depending on which formulation is used; and this will happen to shape-attractor function, too. Figures 1 and 2 show learned parameters from same xy trajectory.

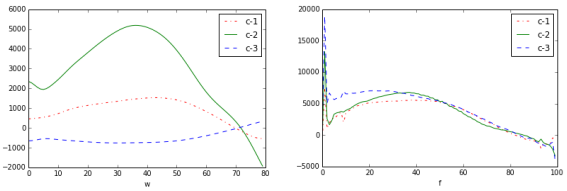

Figure 3. Learned w and f from y coordinate of three *C* letters using original formulation

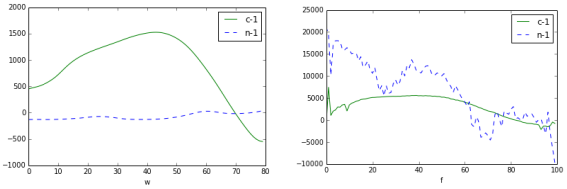

Figure 4. Learned w and f from y coordinate of a *C* and a *N* letter using original formulation

Usually when comparing trajectories it is needed they have same number of samples, and this is typically done re-sampling them before doing comparison. In our case, we are comparing m-DMPs so we need that number of weights w or samples of $f(s)$ are the same for all movements.

The size of w depends on the number N of basis functions, and it could be fixed a-priori for all movements. In case of using the shape-attractor, the function $f(s)$ depends explicitly on the phase variable s which means that number of samples S of this function is equal to number of samples of the learned trajectory, and this is a problem for comparing shape-attractor functions. This could be solved fixing a-priori a number of samples S to generate $f(t)$, that is done using $f(s)$ and $s(t)$ with the fixed S. Therefore, N and S could be considered configuration parameters for movement recognition because they could be fixed a-priori for all movements. However, setting a low N for complex trajectories could cause that different movements seem similar when they are not.

Finally, let's illustrate the results of applying the two formulations to similar movements (Figures 3 and 5) and different ones (Figures 4 and 6). As the weights and shape-attractor function depend on the DMP formulation, as it was explained before, the original and bio-inspired formulations have different properties:

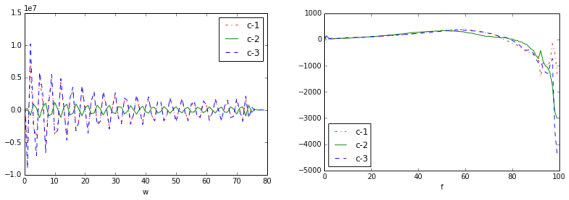

Figure 5. Learned w and f from y coordinate of three *C* letters using bio-inspired formulation

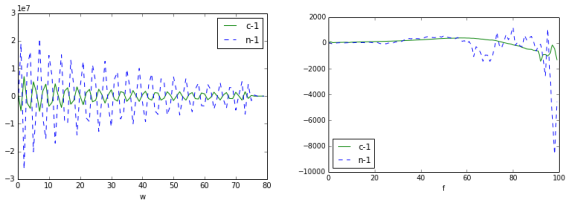

Figure 6. Learned w and f from y coordinate of a *C* and a *N* letter using bio-inspired formulation

- in the original formulation, the weights w of two DMPs that represent the same type of movement could be inverted sometimes. However, this does not occur to the shape-attractor function.
- in the bio-inspired formulation, the weights w are very different for the same type of movement. However, this does not occur to the shape-attractor function.

3.2. Distance-based kernels for m-DMP

Both the shape-attractor function f and weights w are represented as a vector of real values. In order to compare s-DMPs we may use a distance-based kernel for real-values vector as a similarity measure. Then, by aggregating the partial similarities of each s-DMP using an A-average (e.g. arithmetic mean) a global similarity measure can be obtained for an m-DMP [10].

In general, a kernel for real-value vectors $x, y \in R^n$ can be defined as

$$k(x,y) = h\left(\sum_{j=1}^{n} g_j(d_j(x_j,y_j))\right), \quad x_j, y_j \in R \tag{5}$$

for proper choices of h and g functions. For instance, the *Radial Basis Function (RBF)* kernel is defined as

$$k(x,y) = \exp\{-\frac{||x-y||^2}{2\sigma^2}\}, x, y \in R^n, \sigma \not\equiv 0 \in R \tag{6}$$

This particular kernel is obtained by taking $d(x_j, y_j) = |x_j - y_j|$, $g_j(z) = z^2/(2\sigma_j)$ for non-zero σ_j begin a common σ and $h(z) = e^{-z}$.

Another possible choice is the *Truncated Euclidean (TrE)* kernel, which is defined as

$$k(x,y) = \frac{1}{n}\sum_{i=1}^{n} \max\left(0, m - \frac{d_{TrE}(x_i,y_i)}{\sigma}\right), \quad \sigma > 0 \tag{7}$$

where d_i is the *truncated distance* [11], an upper-bounded metric distance defined as:

$$d_{TrE}(x_i, y_i) = \min\{m, |x_i - y_i|\} \tag{8}$$

It is a truncated version of the standard metric in *R*, which can be useful when differences greater than a specified threshold m have to be ignored. In similarity terms, it models situations wherein data examples can become more and more dissimilar until they reach a maximum dissimilarity, from which no more differences are distinguishable.

We consider this distance metric because the dissimilarity of both, the weights w and shape-attractor function f, for two different types of movements may be quite large.

4. Classify Actions from m-DMPs

Our objective is to anchor movements to actions using a bottom-up approach, we want to classify several trajectories that define the same action. The trajectories are defined as m-DMPs and the actions are the types of movements needed by a robot to perform a task.

The experiments were done using IPython with Scikit-Learn [12] and ROS [13]; and the hand-written dataset (HWM) from the LASA[2] was used [14].

4.1. Dataset of hand-written motions

To illustrate our proposed approach a subset of the HWM is used, it is composed by 30 trajectories of 9 different types of movements, see Table 1.

Figure 7. (a) C letter of the dataset, (b) robot used, and (c) trajectory of the robot.

In this dataset the movements are defined by 150 samples on XY plane. And we are defining a m-DMP as the composition of two s-DMPs, one per Cartesian coordinate. Although we are not applying inverse kinematics of the robot during learning because we are working in Cartesian space, we need to apply it to convert the position to joints before sending them to the robot.

Table 1. Number of trajectories per letter in the HWM dataset

C	G	J	N	P	R	S	W	Z
3	3	4	3	3	5	3	3	3

[2]http://lasa.epfl.ch

4.2. *Classification results*

To classify the trajectories in actions we define a feature vector X for each one, and a vector Y for all classes of movements c in the dataset:

$$\begin{aligned} X &= \{x_1, \ldots, x_n, y_1, \ldots, y_n\} \\ Y &= \{c_1, \ldots, c_L\} \end{aligned} \tag{9}$$

where x_i and y_i are the elements of w or f depending on which information is used to classify movements, weights or shape-attractor function. And the value of n will be number of basis functions N or number of samples S depending on that.

We fix $K = 100$ and $D = 2\sqrt{K}$, and we use $N = 80$ basis functions for w and $S = 100$ samples for f. L is always 9 because this is the number of letters in the dataset. The 'leave-one-out' method is used to assess the results of the classifiers because the dataset is small.

First *1-Nearest Neighbors* (1-NN) is used to confirm that f provides better results. Moreover, w are not useful anymore in the bio-inspired formulation. See results in Table 2.

Table 2. Accuracy(%) of the 1-NN using Euclidean distance with 'leave-one-out'

weights		**f(t)**	
original	bio-inspired	original	bio-inspired
0.815	0.333	1.0	0.926

Afterwards, we apply *Support Vector Machines* (SVMs) with *RBF* and *TrE* kernels, which are defined in previous section. The parameters of the kernels are shown in Table 3, and their similarity matrices in Figures 8, 10, 9 and 11. The results, see Table 4, confirm 1-NN conclusions but SVMs do not improve the accuracy of f for the bio-inspired formulation.

Table 3. Parameters used in RBF and TrE kernels

	original		**bio-inspired**	
	RBF	TrE	RBF	TrE
weights	$\sigma = 500$	$m = 750$ $\sigma = 1$	$\sigma = 35000000$	$m = 10000000$ $\sigma = 1$
f(t)	$\sigma = 60000$	$m = 10000$ $\sigma = 1$	$\sigma = 7000$	$m = 350$ $\sigma = 1$

Table 4. Accuracy(%) of the SVM using RBF and TrE kernels with 'leave-one-out'

weights				**f(t)**			
original		bio-inspired		original		bio-inspired	
RBF	**TrE**	**RBF**	**TrE**	**RBF**	**TrE**	**RBF**	**TrE**
0.740	0.111	0.144	0.111	1.0	1.0	0.565	0.778

The results of these experiments confirm our idea that shape-attractor attractor could be used for movement recognition, being a better option than weights in the original formulation and the only one for the bio-inspired formulation.

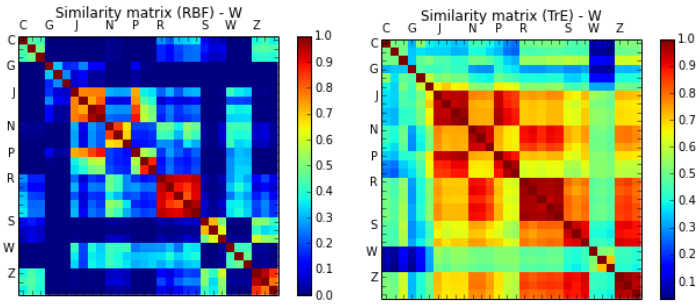

Figure 8. Similarity matrix of w using (a) RBF and (b) TrE from original formulation

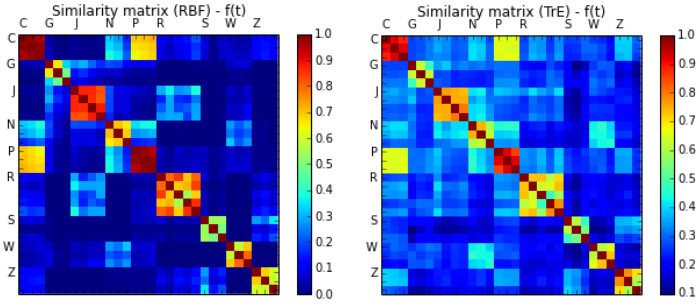

Figure 9. Similarity matrix of f using (a) RBF and (b) TrE from original formulation

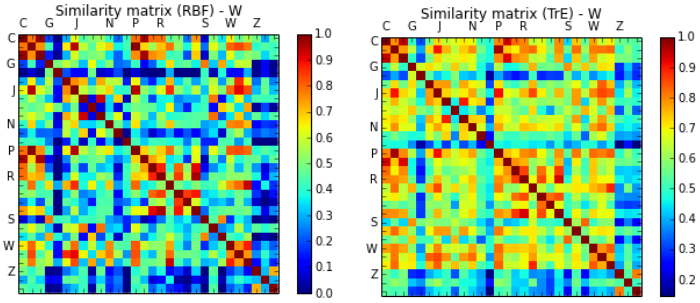

Figure 10. Similarity matrix of w using (a) RBF and (b) TrE from bio-inspired formulation

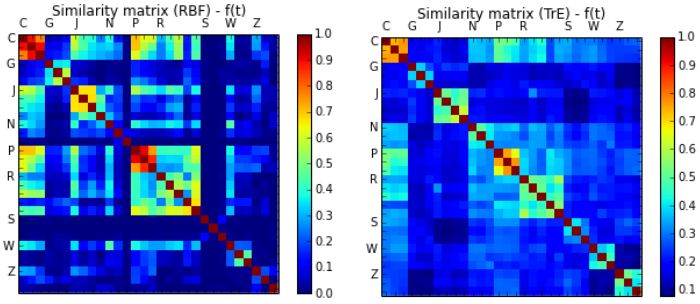

Figure 11. Similarity matrix of f using (a) RBF and (b) TrE from bio-inspired formulation

5. Conclusion

In this paper we have proposed to use the shape-attractor function as feature vector for movement recognition with m-DMPs. This was done after analyzing the two most known DMP formulations, despite weights are used for recognition by most authors working on DMPs. We have also suggested to use distance-based kernels to allow the application of kernel-based learning methods, such as SVMs, to DMPs, in addition to methods directly based on distances, such as k-NN.

These ideas were validated by applying 1-NN and SVM techniques to an existing dataset. For both techniques the results obtained using the shape attractor function were better than those obtained using weights, and the original formulation yielded better performance than the bio-inspired one. As future work, we would like to extend our experiments to larger datasets with more movements and degrees of freedom.

Acknowledgements: This work was partially funded by Spanish research project DPI2013-42458-P.

References

[1] S. Coradeschi and A. Saffiotti, *An Introduction to the Anchoring Problem*, Robotics and Autonomous Systems, vol. 43, no. 2-3, 2003.

[2] A.J. Ijspeert, J. Nakanishi, and S. Schaal, *Learning Attractor Landscapes for Learning Motor Primitives*, Advances in Neural Information Processing Systems 15, 1547-54. MIT Press, 2003.

[3] J. Kober, K. Mülling, O. Kroemer, C. Lampert, B. Schölkopf, and J. Peters, *Movement Templates for Learning of Hitting and Batting*, International Conference on Robotics and Automation (ICRA), 2010.

[4] A. Chella, H. Dindo, and I. Infantino, *A Cognitive Framework for Imitation Learning*, Robotics and Autonomous Systems, vol. 54, no. 5, 2006.

[5] A. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, *Learning Nonlinear Dynamical Systems Models*, Neural Computation, pages 1-33, 2010.

[6] T. Matsubara, S. Hyon, and J. Morimoto, *Learning Parametric Dynamic Movement Primitives from Multiple Demonstrations*, Neural Networks, vol. 24, no. 5, 2011.

[7] M. Prada, A. Remazeilles, A. Koene, and S. Endo, *Dynamic Movement Primitives for Human-Robot Interaction: Comparison with Human Behavioral Observation*, International Conference on Intelligent Robots and Systems (IROS), pages 1168-75, 2013.

[8] H. Hoffmann, P. Pastor, D.H. Park, and S. Schaal, *Biologically-inspired dynamical systems for movement generation: Automatic real-time goal adaptation and obstacle avoidance*, IEEE International Conference on Robotics and Automation (ICRA), pages 2587-2592, 2009.

[9] M. Prada, and A. Remazeilles, *Dynamic Movement Primitives for Human-Robot Interaction*, International Conference on Intelligent Robots and Systems (IROS), pages 1168-1175, 2012.

[10] Ll. Belanche, and A. Tosi, *Averaging of kernel functions*, Neurocomputing, vol. 112, pages 19-25, 2013.

[11] Ll. Belanche, J.L. Vázquez, and M. Vázquez, *Distance-based kernel for real-valued data*, Data Analysis, Machine Learning and Applications, pages 3-10, 2008.

[12] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, vol. 12, pages 2825-2830, 2011.

[13] M. Quigley, K. Conley, B. Gerkey, J. Faust, T.B. Foote, J. Leibs, R. Wheeler, and A.Y. Ng, *ROS: an open-source robot operating system*, in Open-Source Software workshop of the International Conference on Robotics and Automation (ICRA), 2009.

[14] S.M. Khansari Zadeh and A. Billard, *Learning Stable Non-Linear Dynamical Systems with Gaussian Mixture Models*, in IEEE Transaction on Robotics, vol. 27, no. 5, pages 943-957, 2011.

[15] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal, *Learning and Generalization of Motor Skills by Learning from Demonstration*, in IEEE International Conference on Robotics and Automation, 2009.

[16] S. Schaal, J. Peters, J. Nakanishi, and A. Ijspeert, *Learning Movement Primitives*, International Symposium on Robotics Research, Springer Tracts in Advanced Robotics, 2004.